PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
(fraunh01.049)

Applicant:	Luo, et al.	**Confirmation no:**	9357
Application No:	10/526,843	**Group Art Unit:**	2135
Filed:	3/3/2005	**Examiner:**	Gyorfi, Thomas A.

Title: *Protecting mobile code against malicious hosts*

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Brief for a pre-appeal brief conference

Summary of the prosecution

A final rejection in the above patent application issued 9/24/2008. In the final rejection, claims 1-29 were rejected under 35 U.S.C. 103 as obvious over the combination of Monden, "A practical method for watermarking Java programs", hereinafter "Monden", in view of Valdez, "Software DisEngineering: Program hiding architecture and experiments", hereinafter "Valdez". In his rejections, Examiner also applies Official Notice regarding functionality of the Java execution environment, as exemplified by two additional references: Collberg, et al., "A functional taxonomy for software watermarking", hereinafter "Taxonomy", and "How to write Doc comments for the Javadoc tool", hereinafter "Javadoc". The additional references were not made necessary by Applicants' amendment of their claims and consequently, Applicants are protesting Examiner's misuse of Official Notice to add new references to a final rejection. However, even after adding the new references, Examiner has not met his burden under MPEP 2142 of demonstrating that the combination of references used to make the rejection discloses all of the limitations of the claims under rejection. Citations to the above patent application in the following *Traversal* are to its U.S. Patent Application Publication 2006/0026430.

Traversal of the final rejection

What Applicants are claiming

Claim 1 as presently amended is exemplary for what Applicants are claiming. Applicants have added reference numbers and paragraph numbers to the claim to assist the conferees in understanding the claim. The leftmost digits of the reference numbers are figure numbers; thus "113" refers to FIG. 1, while "1503" refers to FIG. 15. The reference numbers and paragraph numbers are not intended to limit the claim:

> **1. (currently amended)** A software object (1503) that contains symbolic names [0017] and is executable in an execution environment (115) for the software object, the execution environment resolving [0100] the symbolic names, the symbolic names including system symbolic names (1403) defined in the execution environment, the execution environment executing in a processor (113), and the software object being contained in a memory device (114) accessible to the processor, and
> the software object comprising:
>
> one or more obfuscated [0011] symbolic names (1505) that correspond to system symbolic names;
>
> a first association (1507) between the obfuscated symbolic names and encrypted forms (1509) of the corresponding system symbolic names; and
>
> a static watermark [0012], [0020] (1504) that has been added to the software object,
>
> the execution environment including a second association (1515) of the encrypted forms 1509 with information (1403, 1407) needed to resolve the corresponding system symbolic names and the execution environment using the first and second associations to resolve the obfuscated symbolic names, and using the static watermark to determine whether the software object has been altered prior to the software object being executed in the program execution environment.

How the software object of claim 1 is produced and used is shown the flowchart of FIG. 16, described at [0107]-[0109].

Limitations which are particularly relevant for the present context are the "first association (1507) between the obfuscated symbolic names and encrypted forms (1509) of the corresponding system symbolic names" in the software object and the "second association (1515) of the encrypted forms 1509 with information (1403, 1407) needed to resolve the corresponding system symbolic names". The steps in FIG. 16 in which the "first association" is made are at 1609 and 1611; the steps in which the "second

association" is made are at 1623-1627. The manner in which the execution environment uses the first and second associations to resolve the obfuscated symbolic names is shown at 1631-1637. It is apparent from Examiner's comment in his Advisory Action that "the second association appears to only undo the obfuscation ... [of] the first association" that Examiner does not understand that the use of the encrypted forms of the unobfuscated system symbolic names in the "first association" and the "second association" is what makes the use of obfuscated system symbolic names in the byte code possible. See [0101] and [0108]-[0109] for details.

What the cited references disclose

Monden

Monden discloses techniques for protecting Java byte codes. At 3.1, Monden mentions obfuscation and its weaknesses for protecting Java programs. At 3.2, he states that an applet can be signed with a digital signature but that this does not protect the class files. At 4, Monden discloses a method of watermarking the Java byte code in which a dummy method that will never be executed is added to the class file. Java code that appears to be valid is then written for the dummy method and the code is compiled. To make the watermark, the dummy method's byte codes are then modified in a fashion such that they remain syntactically correct byte codes. FIGs. 4 and 5 on page 194 show the modification of the byte codes to make the watermark. Monden's watermark is intended to show the source of the byte code. In the example, it contains Monden's name. 4.2 describes how the watermark is decoded. Nowhere is it stated that the decoding is done by the execution environment, and given the purpose of Monden's watermark, there is no reason why the execution environment should decode it.

Examiner takes the discussion of 4 and FIGs. 4 and 5 to show the claim's "first association" and "second association". It is, however, apparent from the foregoing and from a comparison of Monden's FIGs. 4-7 with Applicants' tables 1507 and 1515 that neither figure shows "an "association between the obfuscated symbolic names and encrypted forms (1509) of the corresponding system symbolic names" or an "association

(1515) of the encrypted forms 1509 with information (1403, 1407) needed to resolve the corresponding system symbolic names".

It is further apparent from 4.2 that because Monden's watermark contains no information about the byte code itself, it cannot be used by the execution environment "to determine whether the software object has been altered prior to the software object being executed in the program execution environment", as required by Applicants' claim 1.

Examiner attempts to strengthen the disclosure of Monden with regard to the watermark by adding Javadoc and Taxonomy. Javadoc describes a tool for the automatic documentation of API documentation of Java *source code.* The tool works by adding tags such as `@author` to the source code to indicate information in the source code which is to be extracted from the source code for the API documentation. There is no indication whatever that the tagged material becomes part of the Java byte code, let alone that it becomes a watermark in the byte code. Since the material indicated by `@author` it is not part of the Java byte code, it cannot be used by the execution environment "to determine whether the software object has been altered prior to the software object being executed in the program execution environment", as required by claim 1. Taxonomy merely describes the digital signature mentioned by Monden. The digital signature is not, however, hidden in the byte code and is consequently not a watermark as that term is defined in Applicants' Specification and used in their claims. Thus, neither Monden by itself nor Monden in combination with Javadoc and Taxonomy discloses either Applicants' "first association", Applicants' "second association", or Applicants' watermark that is used by the execution environment "to determine whether the software object has been altered prior to the software object being executed in the program execution environment."

Valdez

Believing as he does that Monden only lacks encryption, Examiner cites Valdez for the proposition that encryption can be used to obscure symbolic information (Final Office action, page 6, top). Given the failure of Monden, Javadoc, and Taxonomy to disclose

anything like the "first association", "second association" or "watermark" set forth in claim 1, more is required of Valdez than merely showing that encryption can be used to obscure symbolic information if Examiner's rejection is to stand. What Applicants are claiming in claim 1 is not just encryption, but rather a first association between obfuscated system symbolic names and encrypted forms of the corresponding system symbolic names and a second association of the encrypted forms with information needed to resolve the system symbolic names. Claim 1 thus requires encryption of individual system symbolic names. Valdez, however, encrypts blocks of his code, not symbolic names:

> After applying the noise and substitution primitives, the processed sections [of TCL code] are decomposed into blocks of some finite size ... from these blocks, blocks are randomly selected for encryption ... (Valdez, p. 386, lines 16-18)

Because Valdez encrypts blocks and not individual syntactic elements in the code, he cannot encrypt individual symbolic names, and because he cannot do so, there can be nothing like Applicants' "first association" or "second association" in his system.

The patentability of Applicants' claims over the references

Because the combined references do not disclose either claim 1's "first association" or its "second association" or its "watermark", Examiner's rejection of claim 1 under 35 U.S.C. 103 is without basis. As the Conferees will immediately see, the "first association" and "second association" that are used "to resolve the obfuscated symbolic names" are limitations of each of Applicants' independent claims 1, 6, 14, and 22, and consequently all of these claims and all of the claims dependent on these claims are patentable over the references. Since that is the case, Applicants respectfully request that the Conferees allow this application or failing that, reopen prosecution. The required *Request for a pre-appeal brief conference* accompanies this brief along with the required *Notice of appeal* and the requisite fee, as well as the fee for a two-month extension of time. Please charge any additional fees required for the *Request* or refund any overpayments to deposit account number 501315.

Respectfully submitted,

/Gordon E. Nelson/

Attorney of record,
Gordon E. Nelson
57 Central St., P.O. Box 782
Rowley, MA, 01969,
Registration number 30,093
Voice: (978) 948-7632
Fax: (866) 723-0359

2/23/2009
Date